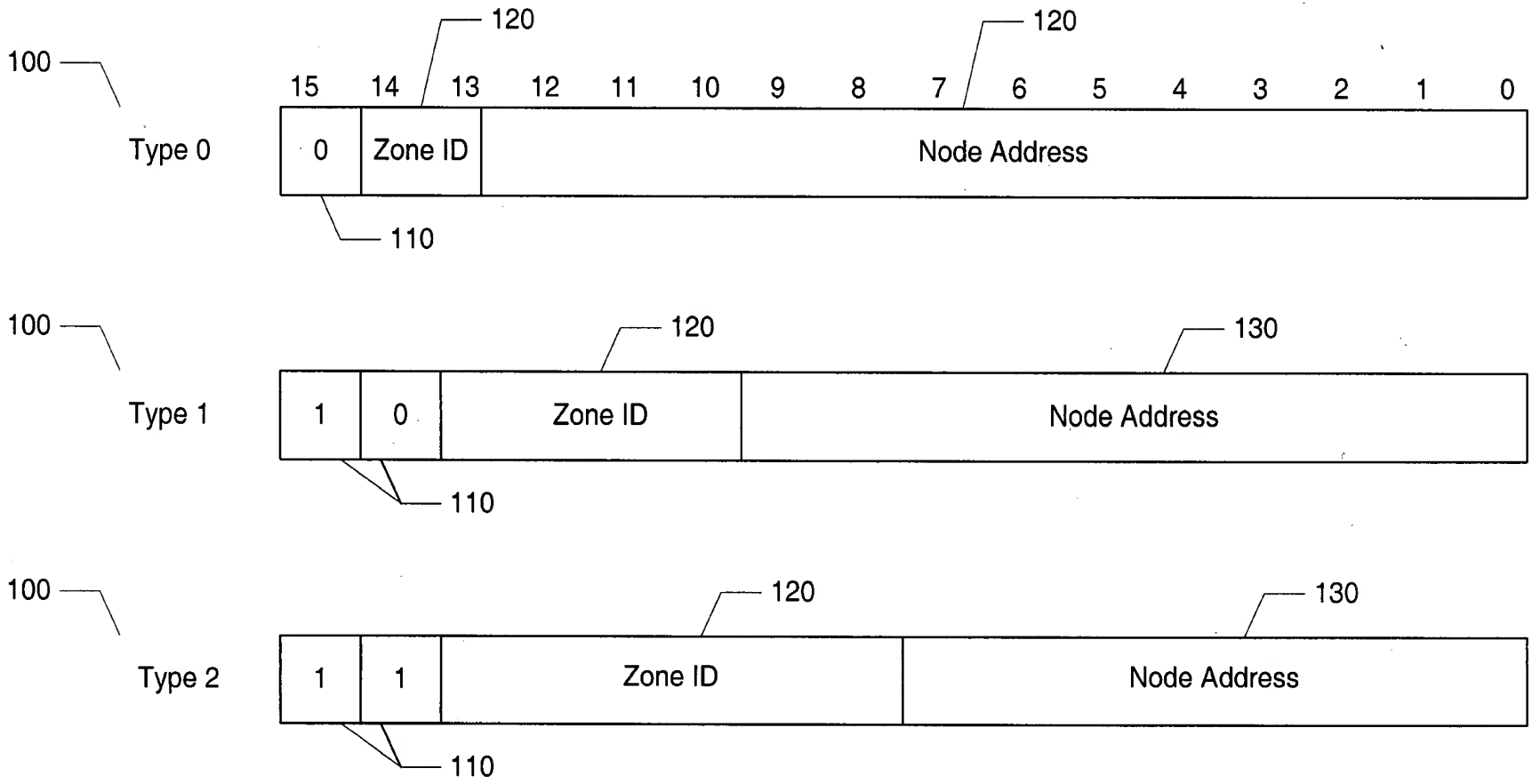

Fig. 1

JC929 U.S. PTO
09/751763
12/30/00

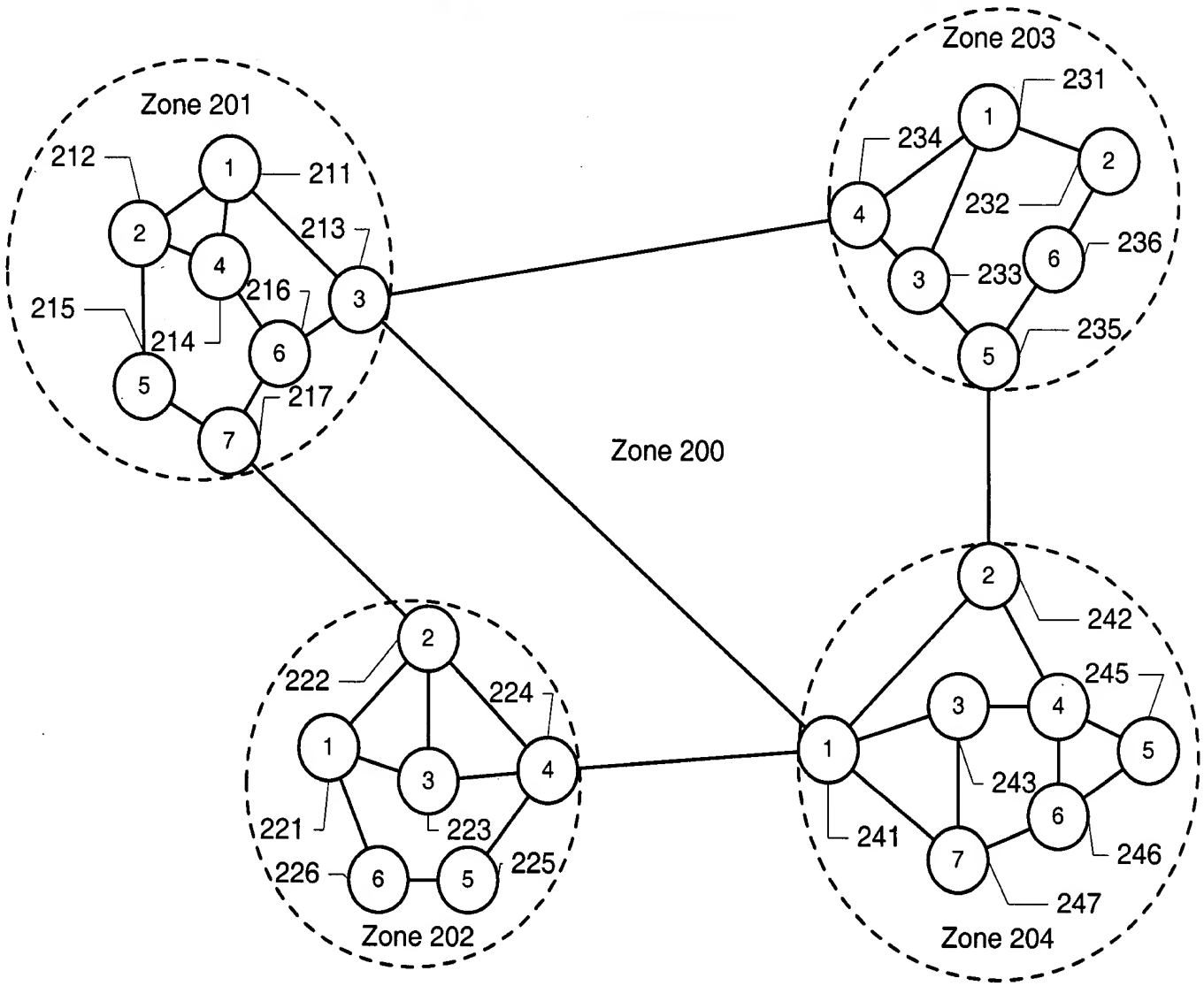
Zone 201
212
211
213
215
214
216
217
Zone 203
231
234
232
236
233
235
Zone 200
Zone 202
222
224
221
223
226
225
Zone 204
242
245
243
241
246
247

Fig. 2

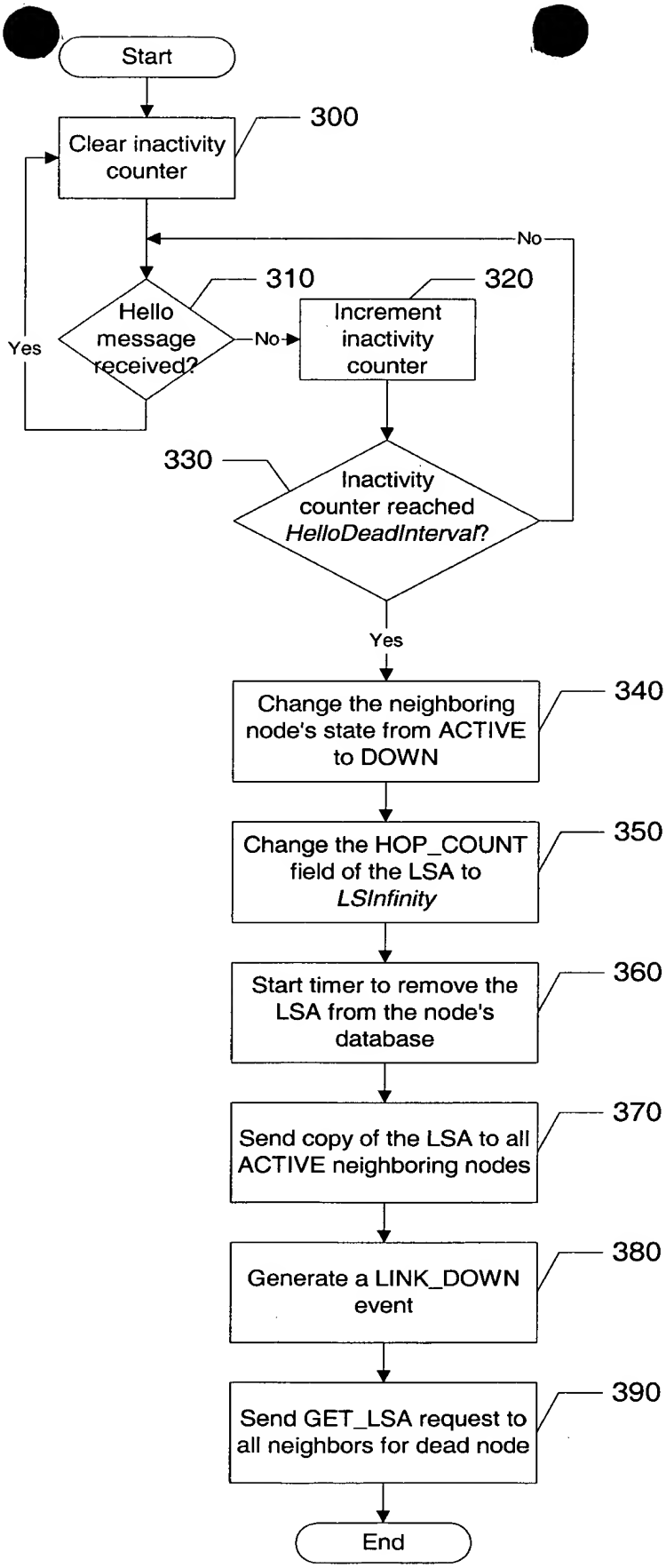
Start
Clear inactivity counter
300
No
310
320
Hello message received?
Yes
No
Increment inactivity counter
330
Inactivity counter reached HelloDeadInterval?
Yes
Change the neighboring node's state from ACTIVE to DOWN
340
Change the HOP_COUNT field of the LSA to LSInfinity
350
Start timer to remove the LSA from the node's database
360
Send copy of the LSA to all ACTIVE neighboring nodes
370
Generate a LINK_DOWN event
380
Send GET_LSA request to all neighbors for dead node
390
End

Fig. 3

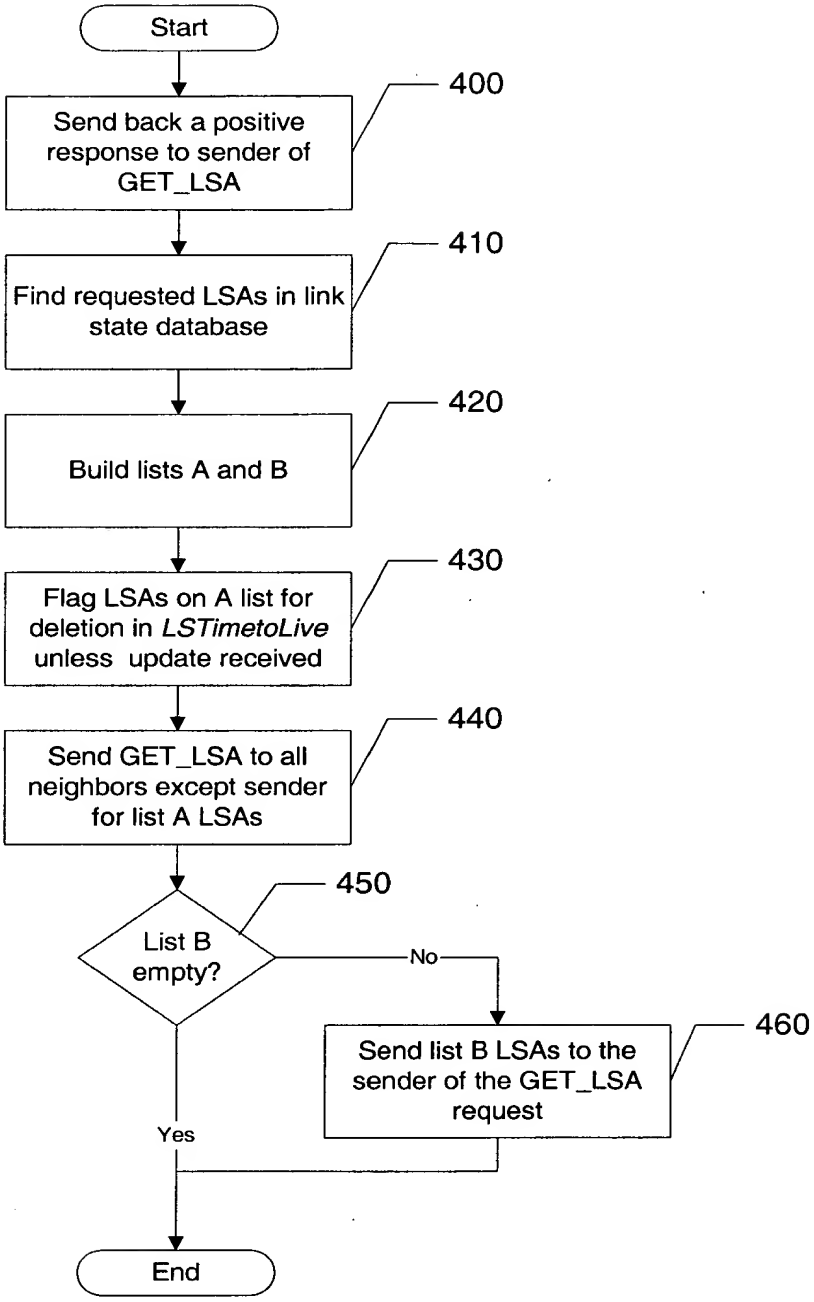
Start
Send back a positive response to sender of GET_LSA
400
Find requested LSAs in link state database
410
Build lists A and B
420
Flag LSAs on A list for deletion in *LSTimetoLive* unless update received
430
Send GET_LSA to all neighbors except sender for list A LSAs
440
List B empty?
450
No
Send list B LSAs to the sender of the GET_LSA request
460
Yes
End

Fig. 4

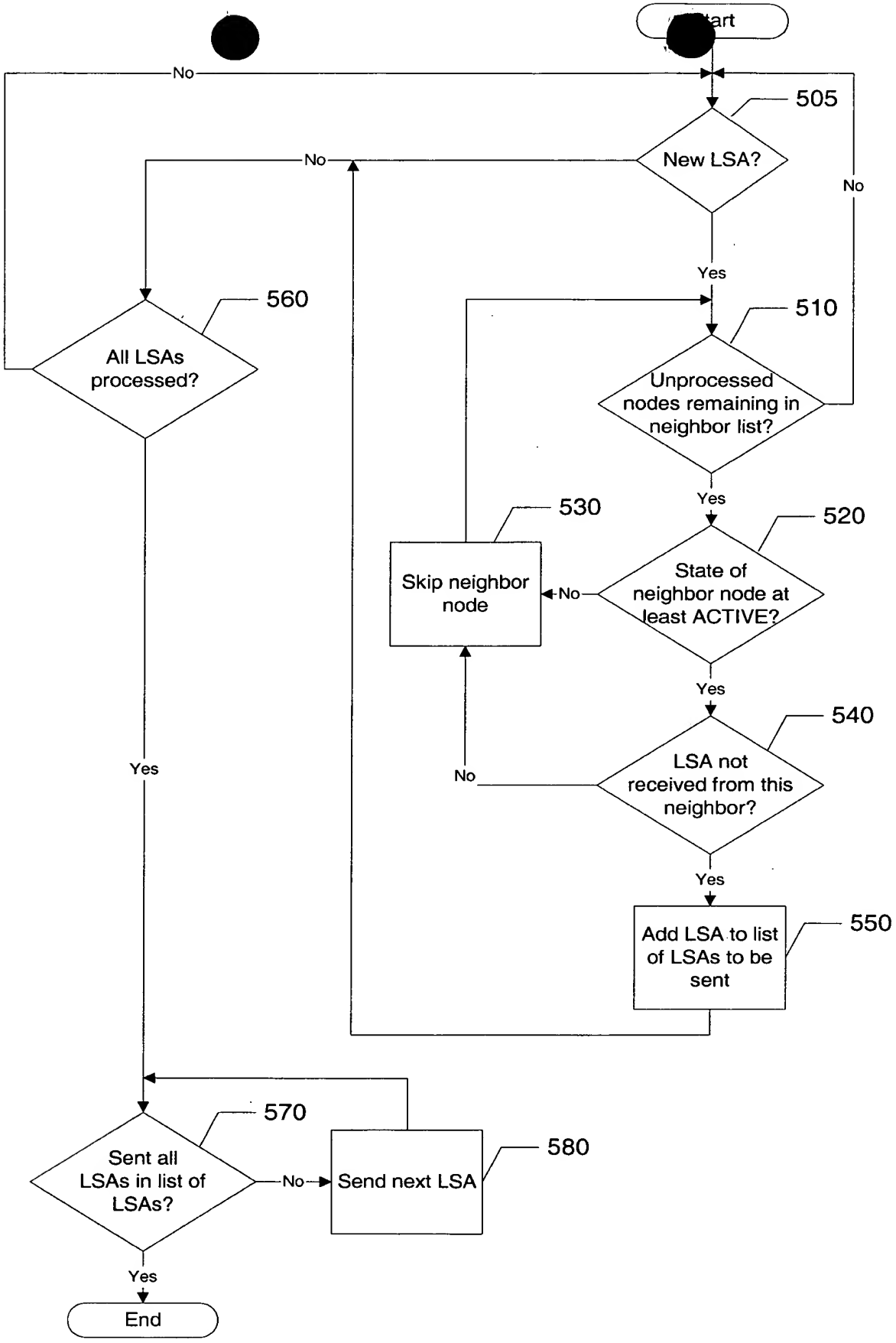
Start
505
New LSA?
No
Yes
No
510
Unprocessed nodes remaining in neighbor list?
Yes
520
State of neighbor node at least ACTIVE?
No
530
Skip neighbor node
Yes
540
LSA not received from this neighbor?
No
Yes
550
Add LSA to list of LSAs to be sent
560
All LSAs processed?
No
Yes
570
Sent all LSAs in list of LSAs?
No
580
Send next LSA
Yes
End

Fig. 5

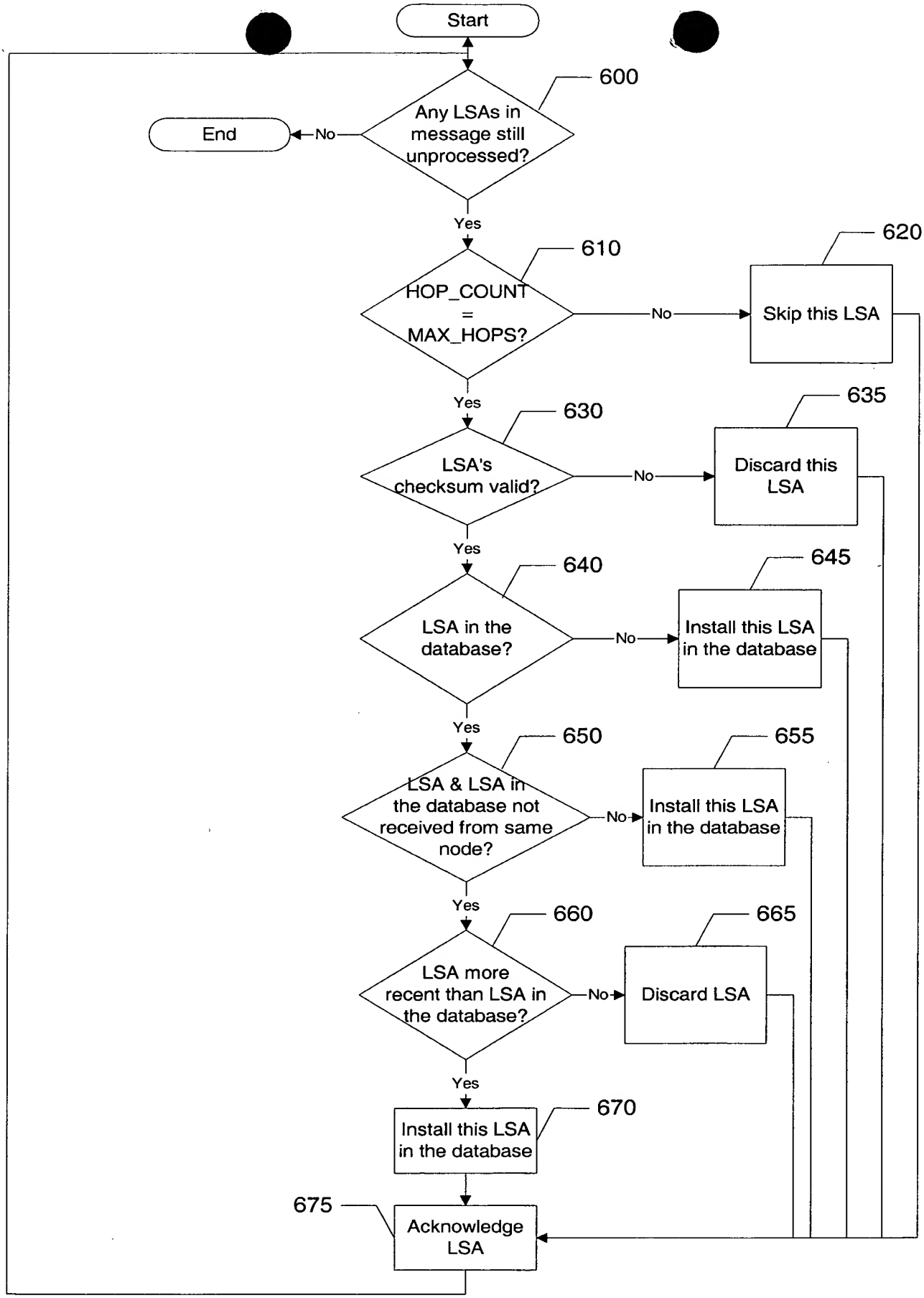
Start
600
Any LSAs in message still unprocessed?
No
End
Yes
610
HOP_COUNT = MAX_HOPS?
No
620
Skip this LSA
Yes
630
LSA's checksum valid?
No
635
Discard this LSA
Yes
640
LSA in the database?
No
645
Install this LSA in the database
Yes
650
LSA & LSA in the database not received from same node?
No
655
Install this LSA in the database
Yes
660
LSA more recent than LSA in the database?
No
665
Discard LSA
Yes
670
Install this LSA in the database
675
Acknowledge LSA

Fig. 6

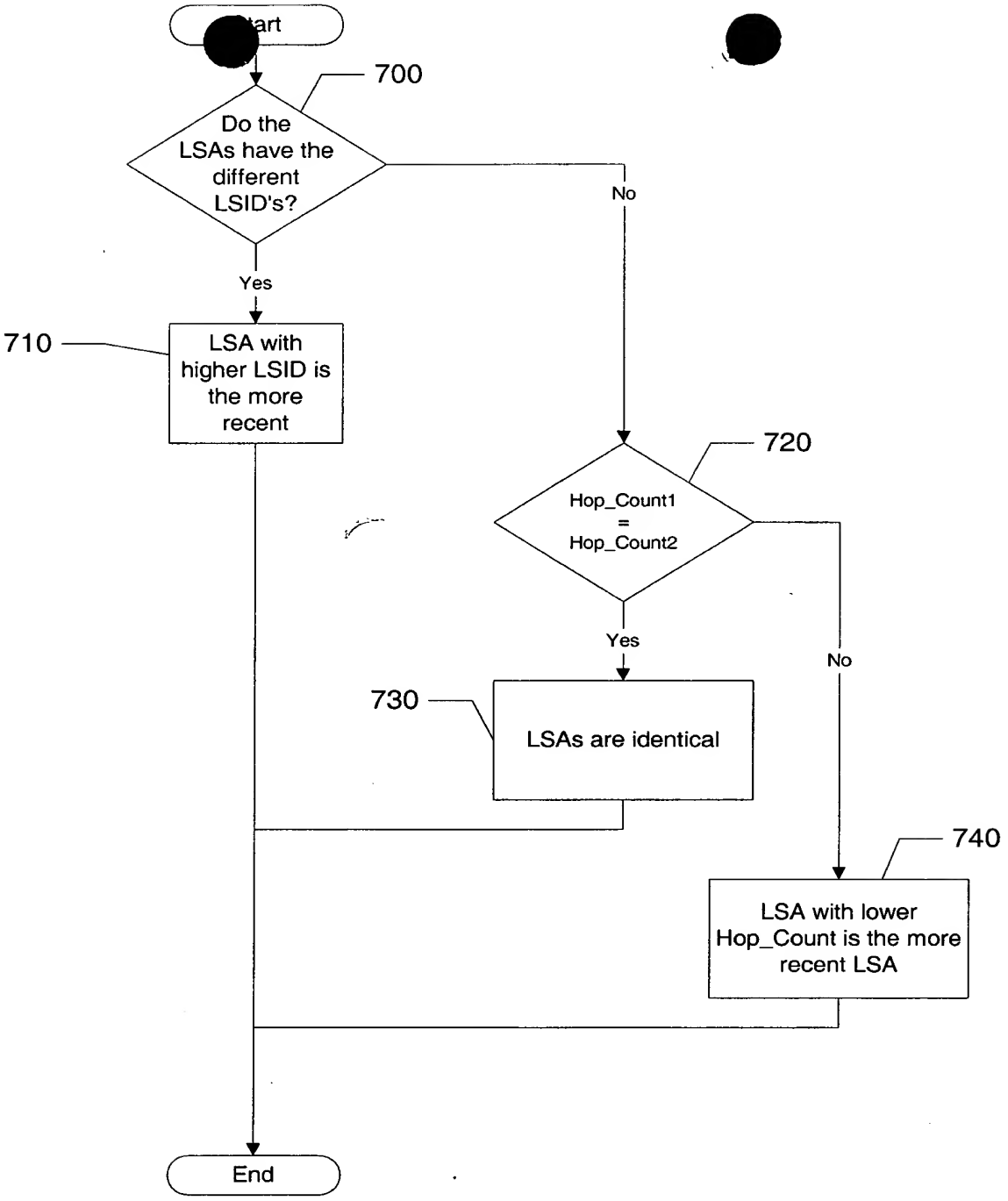
Start
700
Do the LSAs have the different LSID's?
No
Yes
710
LSA with higher LSID is the more recent
720
Hop_Count1 = Hop_Count2
Yes
No
730
LSAs are identical
740
LSA with lower Hop_Count is the more recent LSA
End

Fig. 7

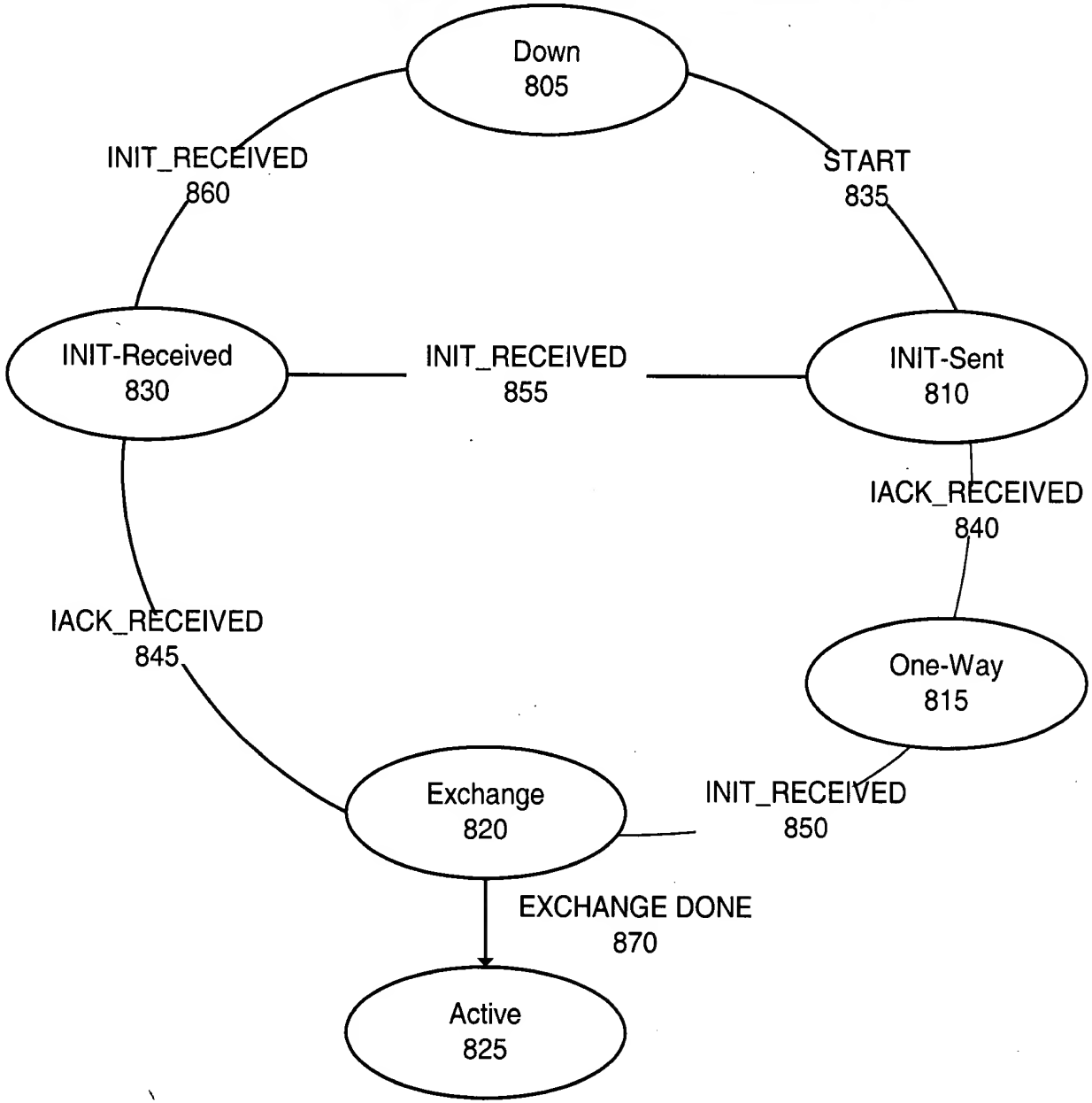
Down
805
INIT_RECEIVED
860
START
835
INIT-Received
830
INIT_RECEIVED
855
INIT-Sent
810
IACK_RECEIVED
840
IACK_RECEIVED
845
One-Way
815
Exchange
820
INIT_RECEIVED
850
EXCHANGE DONE
870
Active
825

Fig. 8

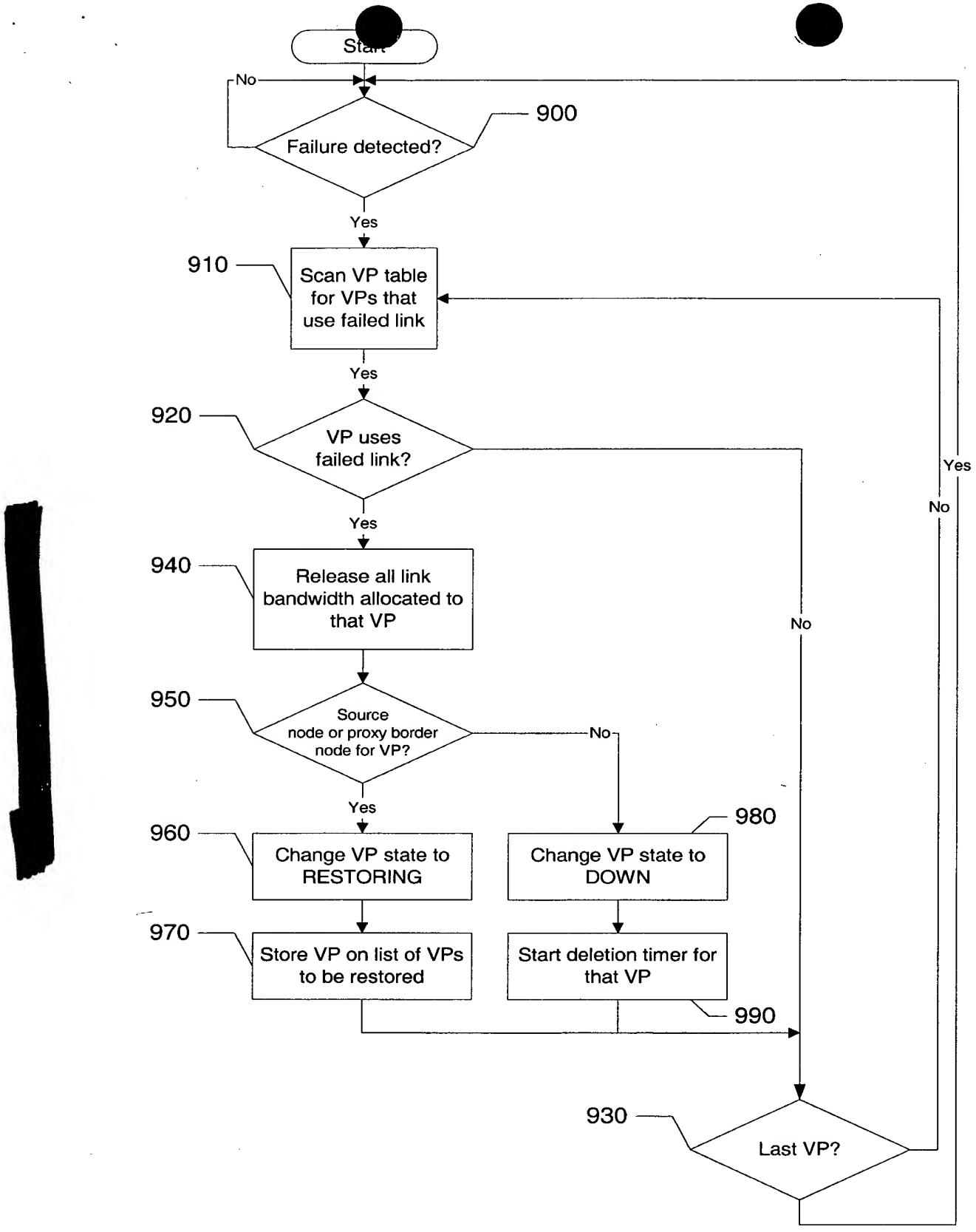
Start
No
900
Failure detected?
Yes
910
Scan VP table for VPs that use failed link
Yes
920
VP uses failed link?
Yes
No
Yes
940
Release all link bandwidth allocated to that VP
No
950
Source node or proxy border node for VP?
No
Yes
960
Change VP state to RESTORING
980
Change VP state to DOWN
970
Store VP on list of VPs to be restored
Start deletion timer for that VP
990
930
Last VP?

Fig. 9

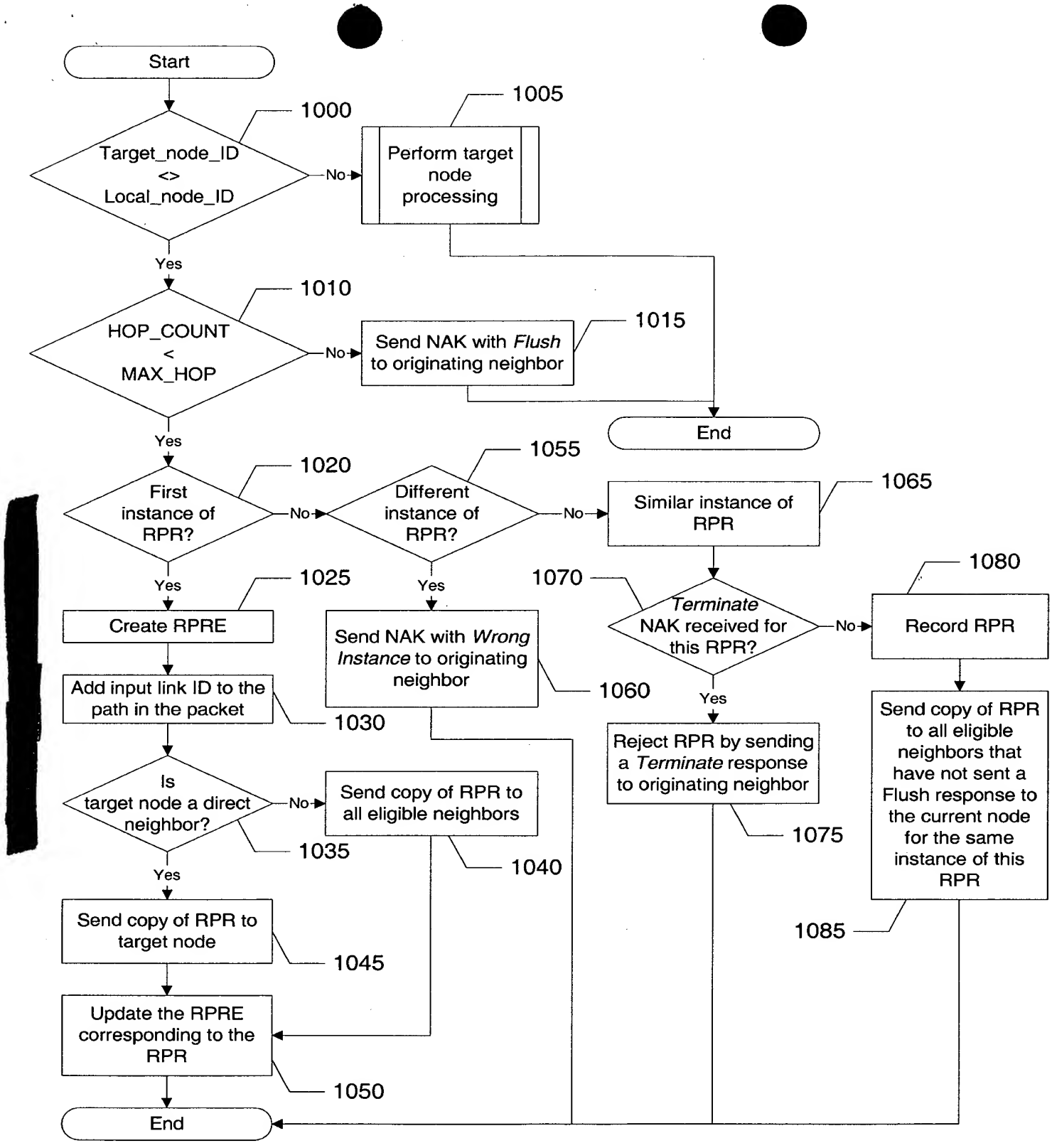
Start
1000
Target_node_ID <> Local_node_ID
No
1005
Perform target node processing
Yes
1010
HOP_COUNT < MAX_HOP
No
1015
Send NAK with *Flush* to originating neighbor
End
Yes
1020
First instance of RPR?
No
1055
Different instance of RPR?
No
1065
Similar instance of RPR
Yes
1025
Create RPRE
Yes
Send NAK with *Wrong Instance* to originating neighbor
1060
1070
Terminate NAK received for this RPR?
No
1080
Record RPR
Add input link ID to the path in the packet
1030
Yes
Reject RPR by sending a *Terminate* response to originating neighbor
1075
Send copy of RPR to all eligible neighbors that have not sent a Flush response to the current node for the same instance of this RPR
1085
Is target node a direct neighbor?
No
Send copy of RPR to all eligible neighbors
1035
1040
Yes
Send copy of RPR to target node
1045
Update the RPRE corresponding to the RPR
1050
End

Fig. 10

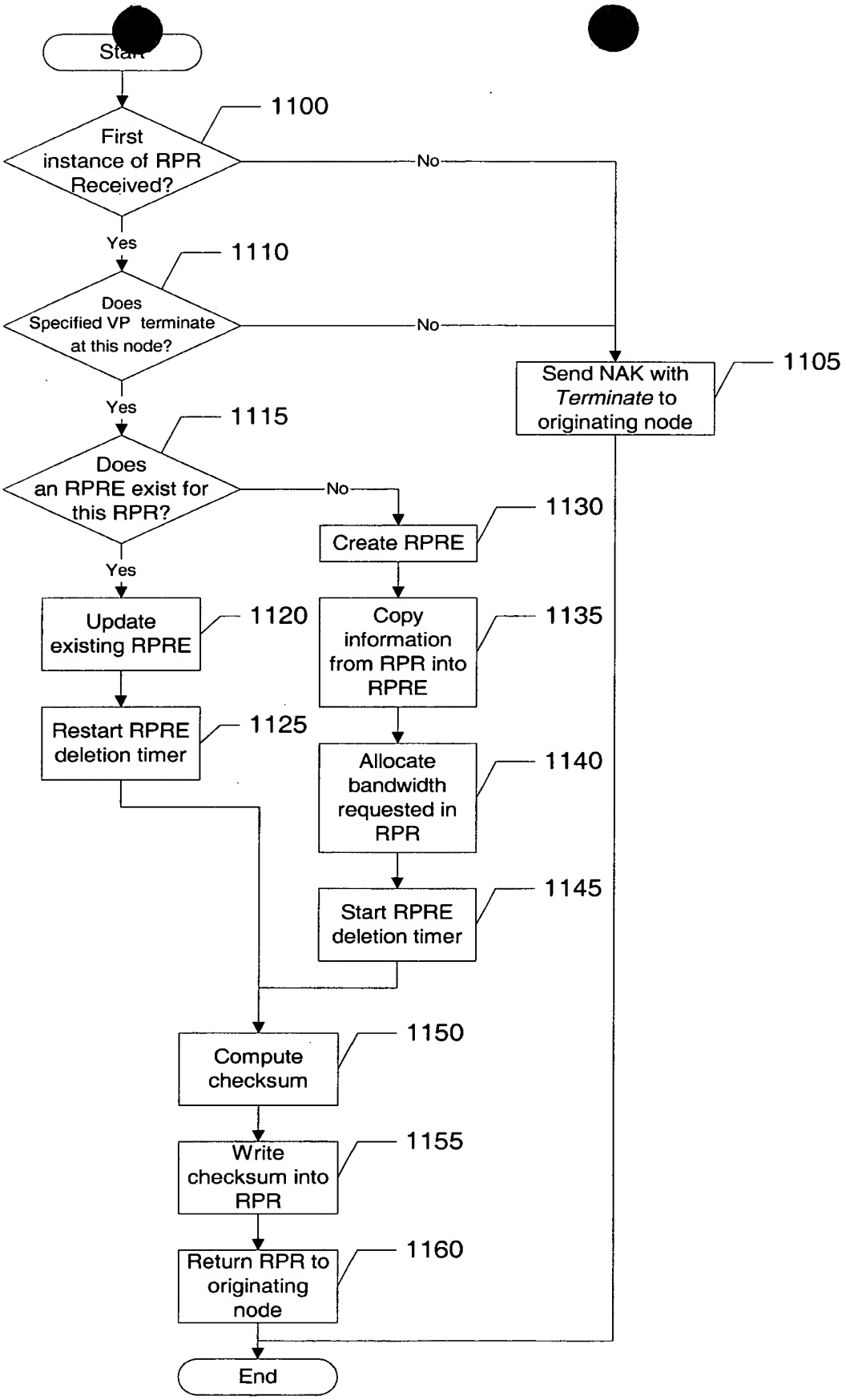
Start
1100
First instance of RPR Received?
No
Yes
1110
Does Specified VP terminate at this node?
No
Yes
1105
Send NAK with *Terminate* to originating node
1115
Does an RPRE exist for this RPR?
No
Yes
1130
Create RPRE
1120
Update existing RPRE
1135
Copy information from RPR into RPRE
1125
Restart RPRE deletion timer
1140
Allocate bandwidth requested in RPR
1145
Start RPRE deletion timer
1150
Compute checksum
1155
Write checksum into RPR
1160
Return RPR to originating node
End

Fig. 11

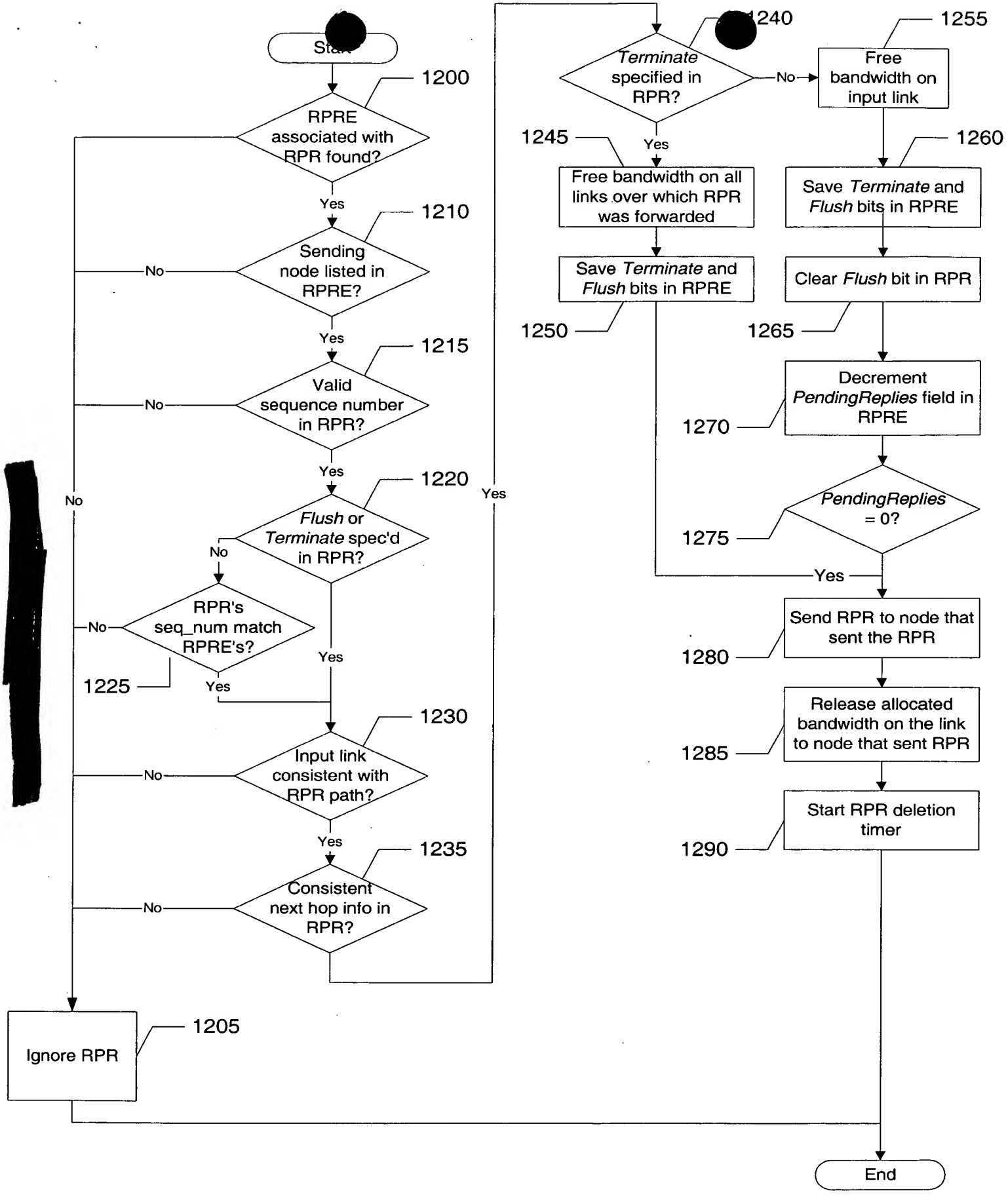
Start
1200
RPRE associated with RPR found?
Yes
No
1210
Sending node listed in RPRE?
Yes
No
1215
Valid sequence number in RPR?
Yes
No
1220
Flush or Terminate spec'd in RPR?
No
Yes
RPR's seq_num match RPRE's?
No
Yes
1225
1230
Input link consistent with RPR path?
No
Yes
1235
Consistent next hop info in RPR?
No
Yes
Ignore RPR
1205
1240
Terminate specified in RPR?
No
Yes
1245
Free bandwidth on all links over which RPR was forwarded
Save Terminate and Flush bits in RPRE
1250
1255
Free bandwidth on input link
1260
Save Terminate and Flush bits in RPRE
Clear Flush bit in RPR
1265
Decrement PendingReplies field in RPRE
1270
PendingReplies = 0?
1275
Yes
Send RPR to node that sent the RPR
1280
Release allocated bandwidth on the link to node that sent RPR
1285
Start RPR deletion timer
1290
End

Fig. 12

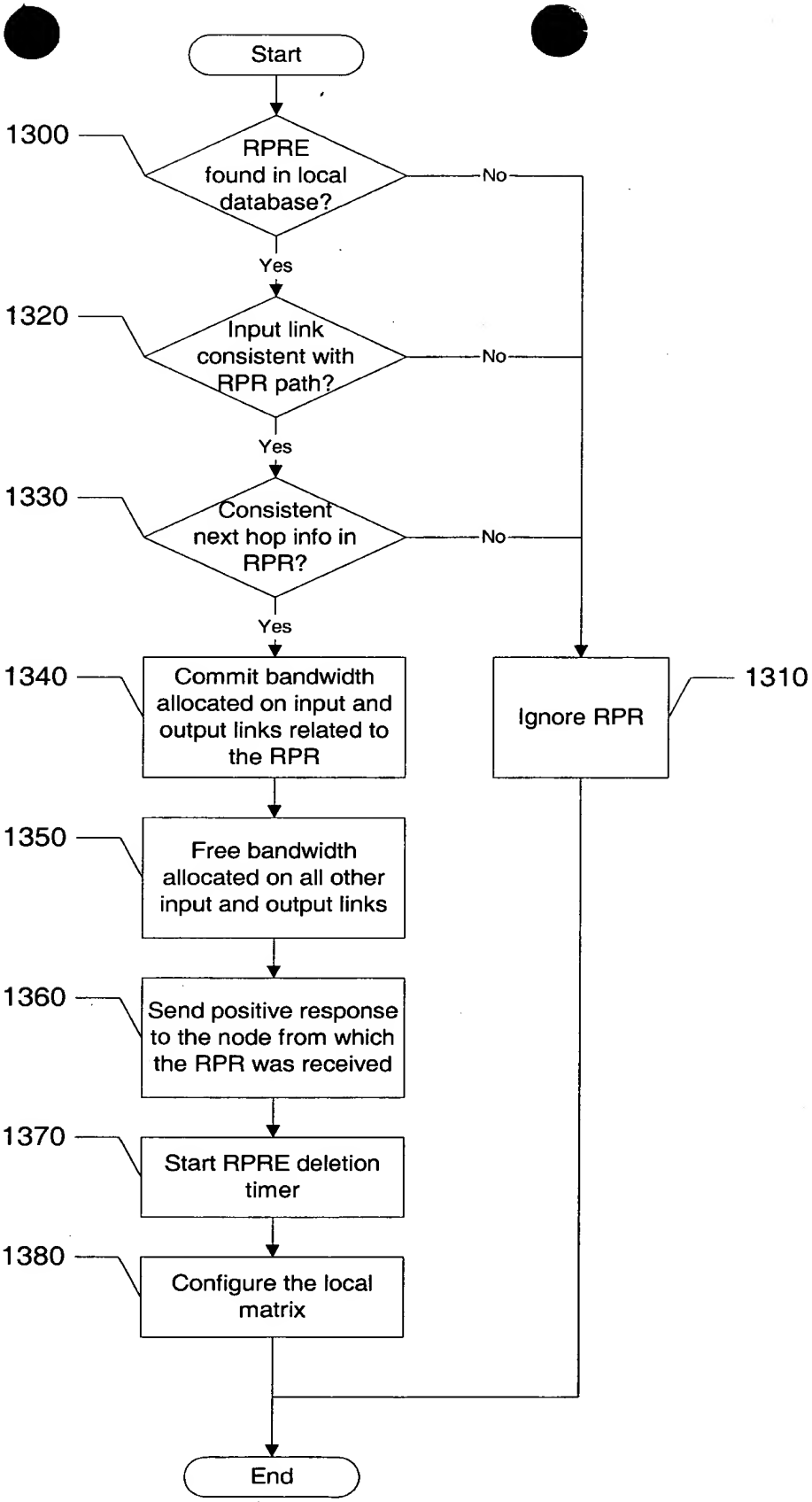
Start
1300
RPRE found in local database?
No
Yes
1320
Input link consistent with RPR path?
No
Yes
1330
Consistent next hop info in RPR?
No
Yes
1340
Commit bandwidth allocated on input and output links related to the RPR
1310
Ignore RPR
1350
Free bandwidth allocated on all other input and output links
1360
Send positive response to the node from which the RPR was received
1370
Start RPRE deletion timer
1380
Configure the local matrix
End

Fig. 13

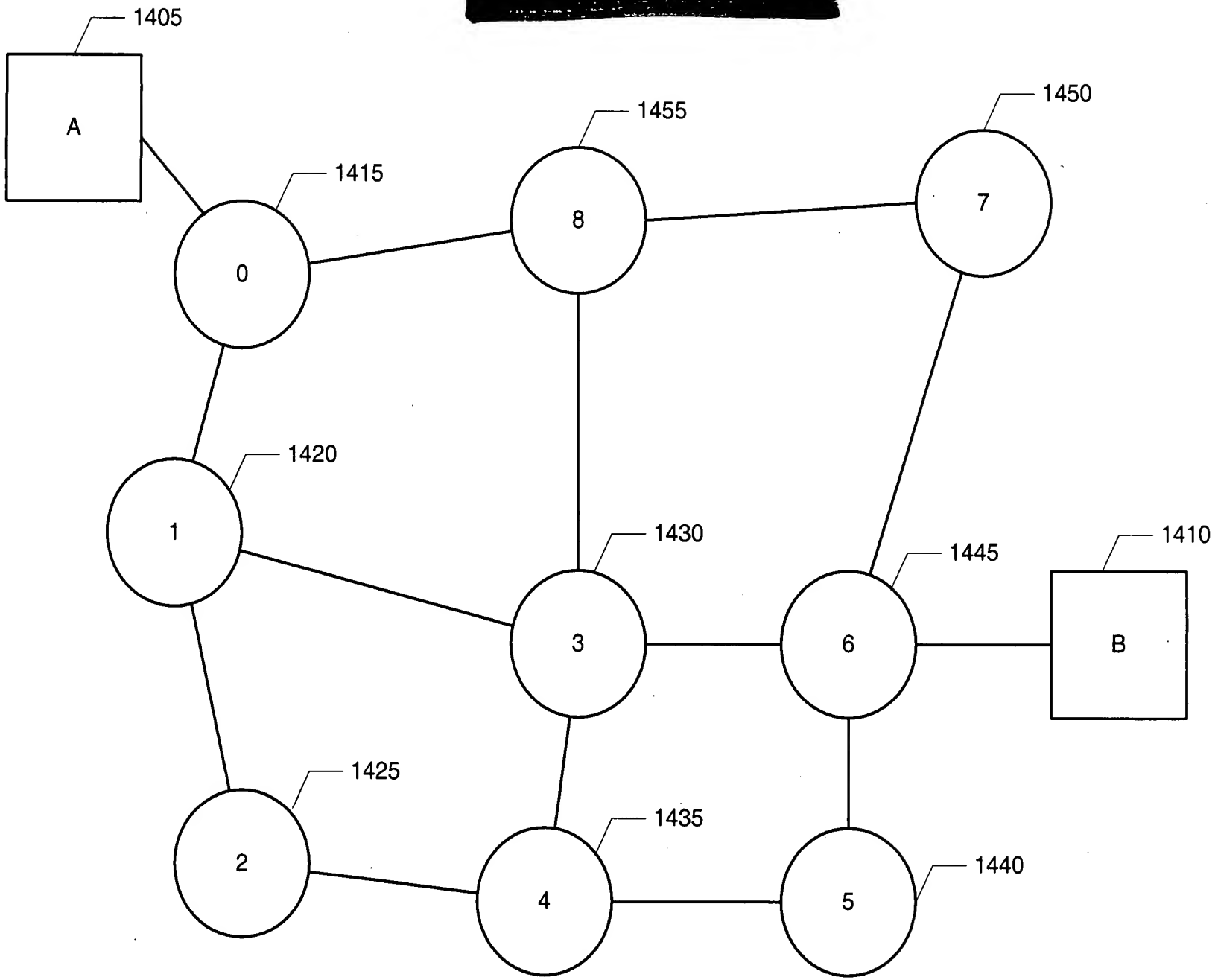
1405
A
1455
1450
1415
8
7
0
1420
1
1430
1445
1410
3
6
B
1425
1435
1440
2
4
5

Fig. 14

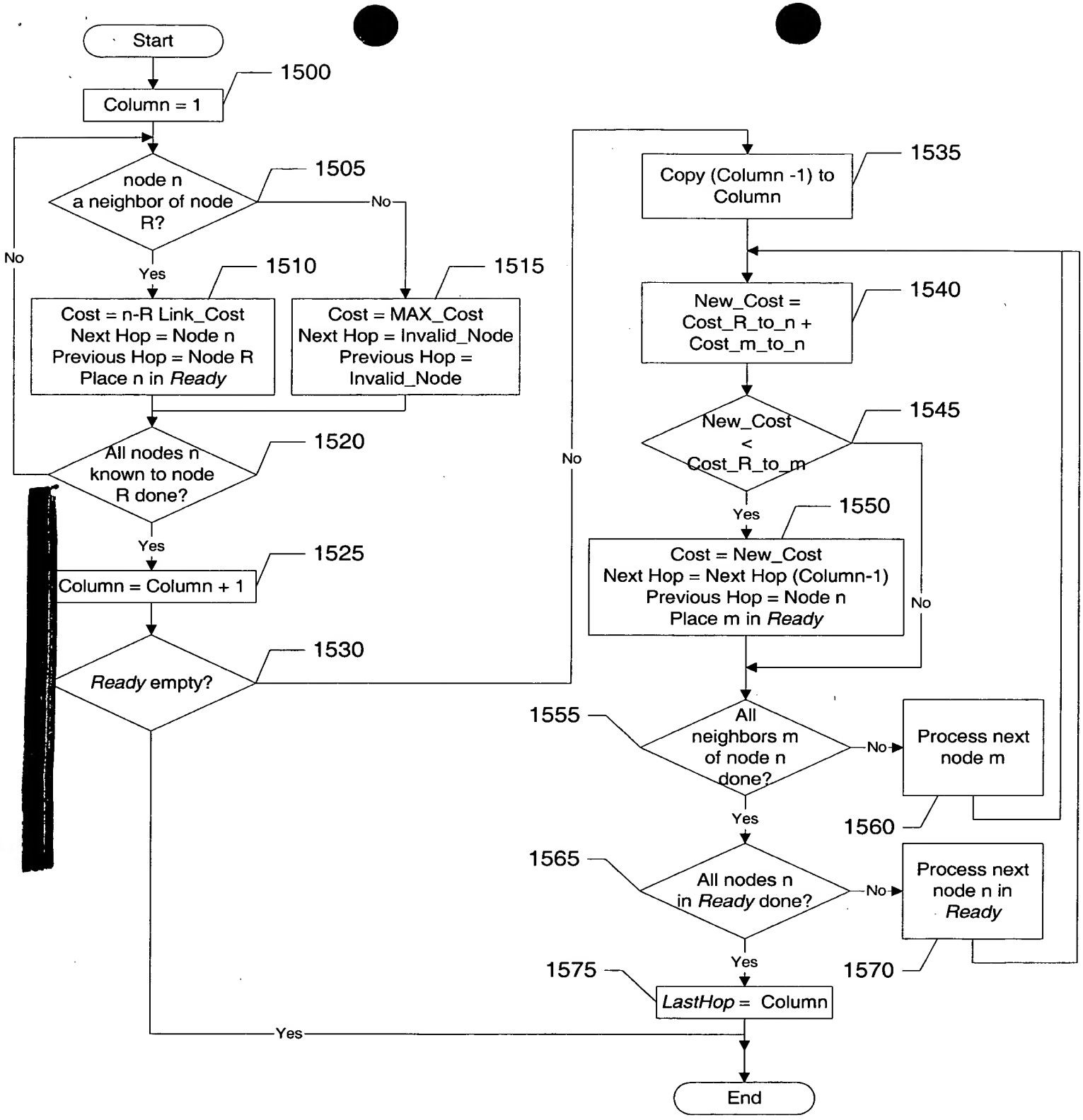
Start
1500
Column = 1
1505
node n a neighbor of node R?
No
Yes
1510
Cost = n-R Link_Cost
Next Hop = Node n
Previous Hop = Node R
Place n in Ready
1515
Cost = MAX_Cost
Next Hop = Invalid_Node
Previous Hop = Invalid_Node
1520
All nodes n known to node R done?
No
Yes
1525
Column = Column + 1
1530
Ready empty?
No
Yes
1535
Copy (Column -1) to Column
1540
New_Cost = Cost_R_to_n + Cost_m_to_n
1545
New_Cost < Cost_R_to_m
Yes
No
1550
Cost = New_Cost
Next Hop = Next Hop (Column-1)
Previous Hop = Node n
Place m in Ready
1555
All neighbors m of node n done?
No
Yes
Process next node m
1560
1565
All nodes n in Ready done?
No
Yes
Process next node n in Ready
1570
1575
LastHop = Column
End

Fig. 15

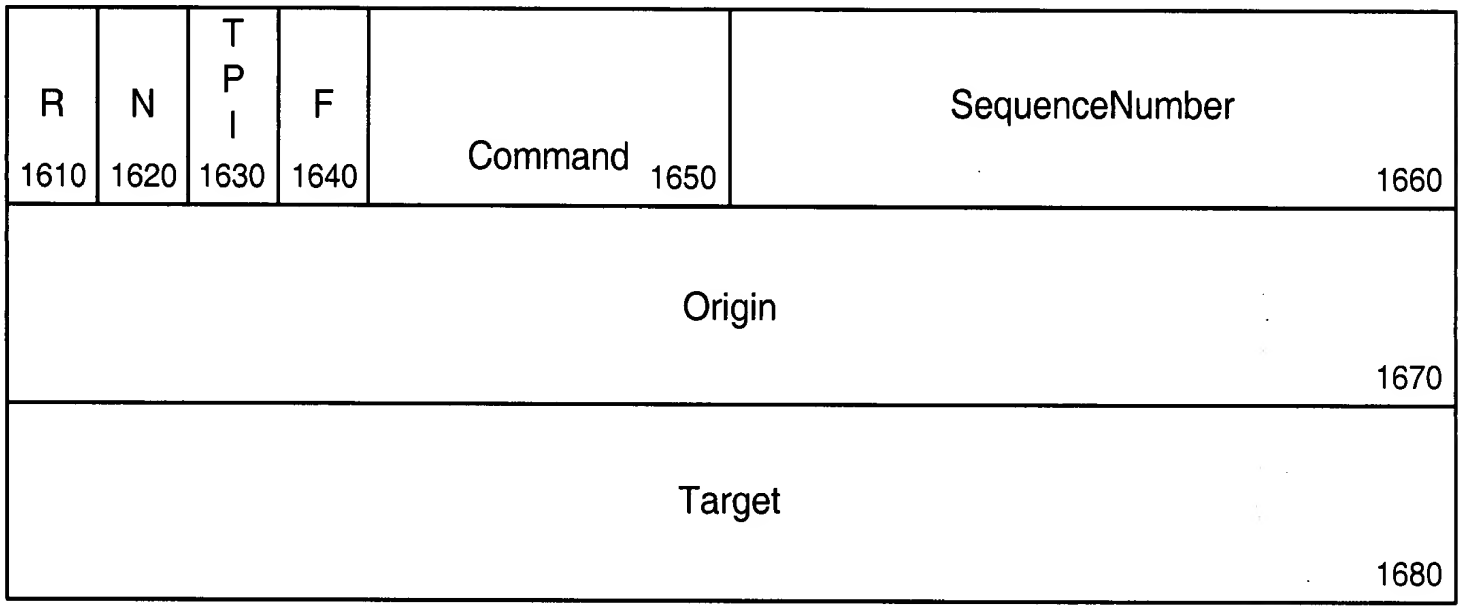
1600
R
1610
N
1620
TPI
1630
F
1640
Command
1650
SequenceNumber
1660
Origin
1670
Target
1680

Fig. 16

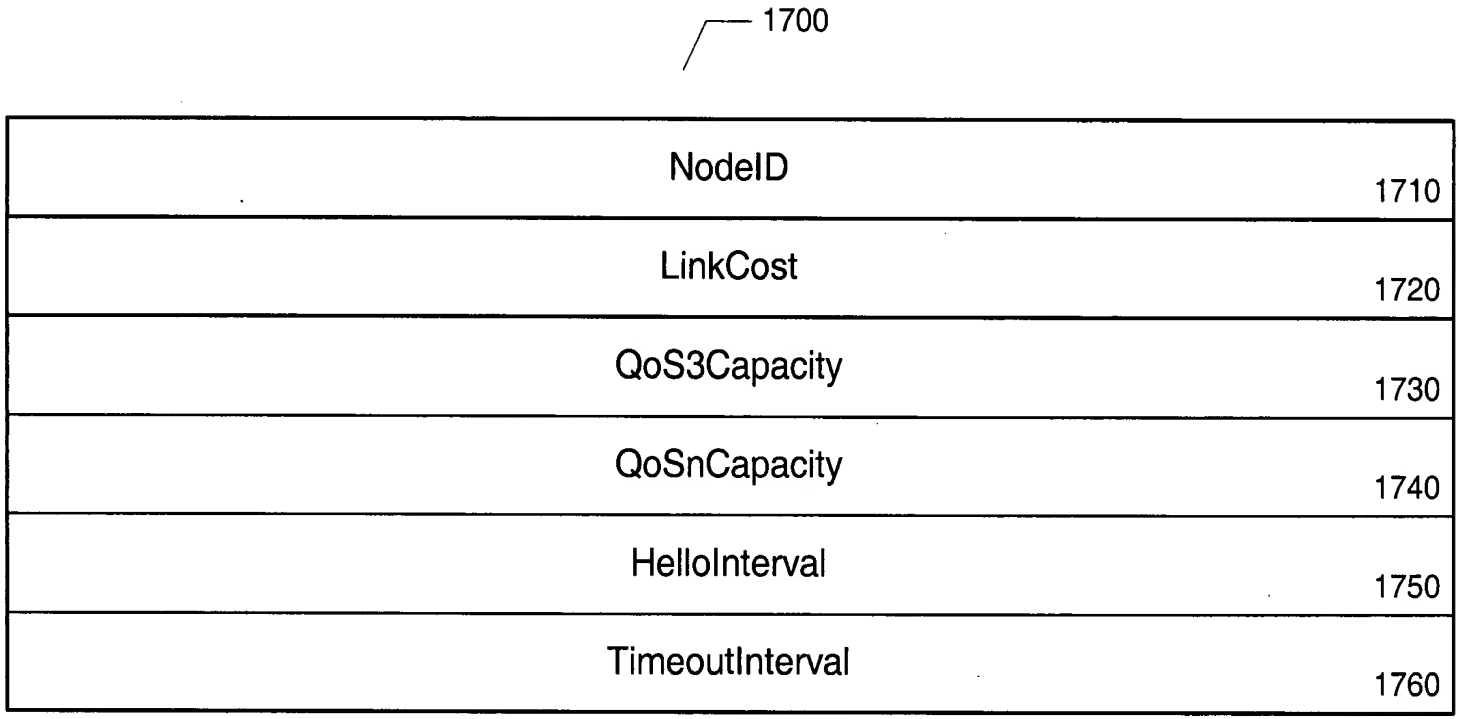
1700
NodeID
1710
LinkCost
1720
QoS3Capacity
1730
QoSnCapacity
1740
HelloInterval
1750
TimeoutInterval
1760

Fig. 17

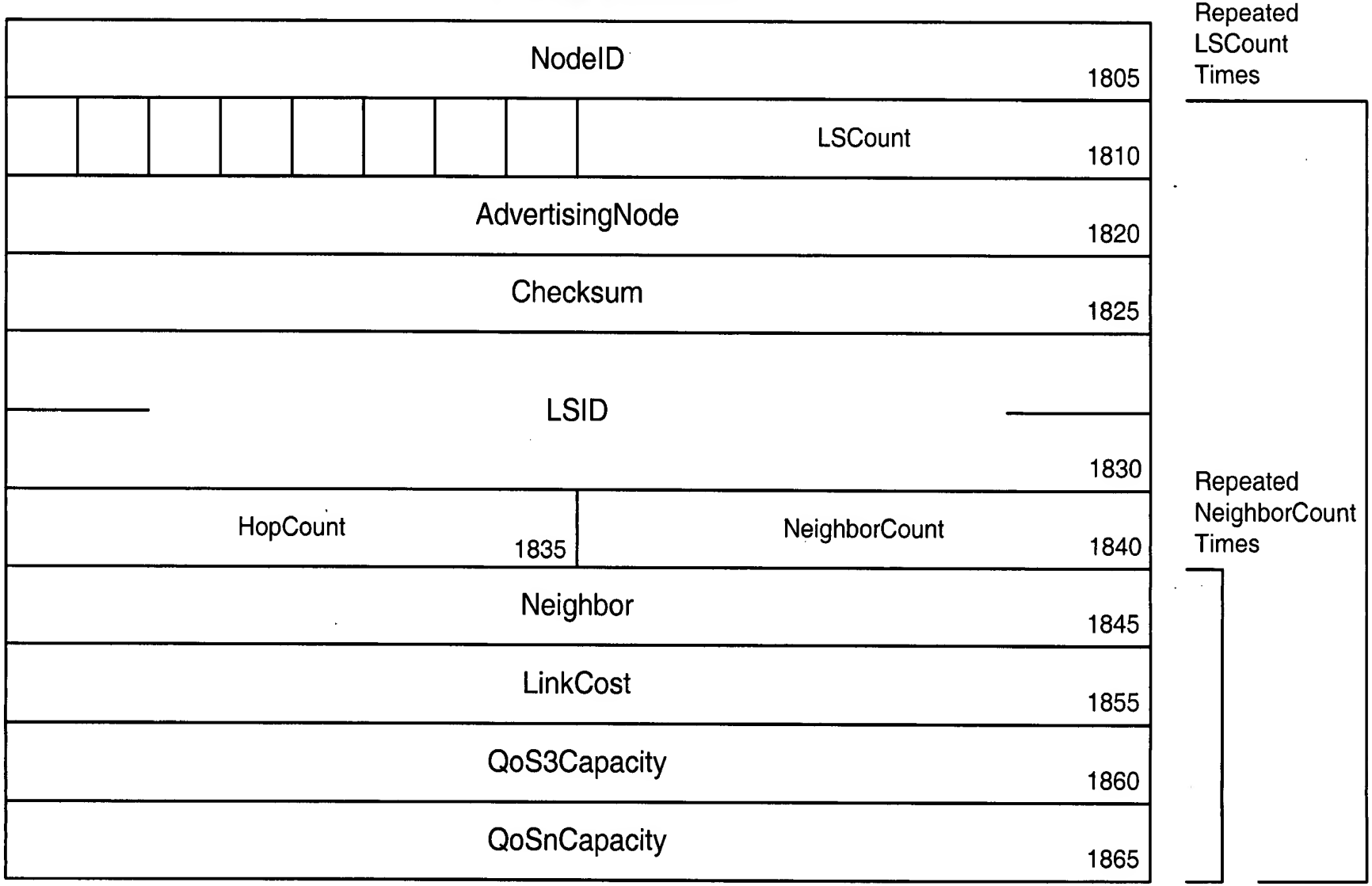
NodeID
1805
LSCount
1810
AdvertisingNode
1820
Checksum
1825
LSID
1830
HopCount
1835
NeighborCount
1840
Neighbor
1845
LinkCost
1855
QoS3Capacity
1860
QoSnCapacity
1865
Repeated LSCount Times
Repeated NeighborCount Times

Fig. 18

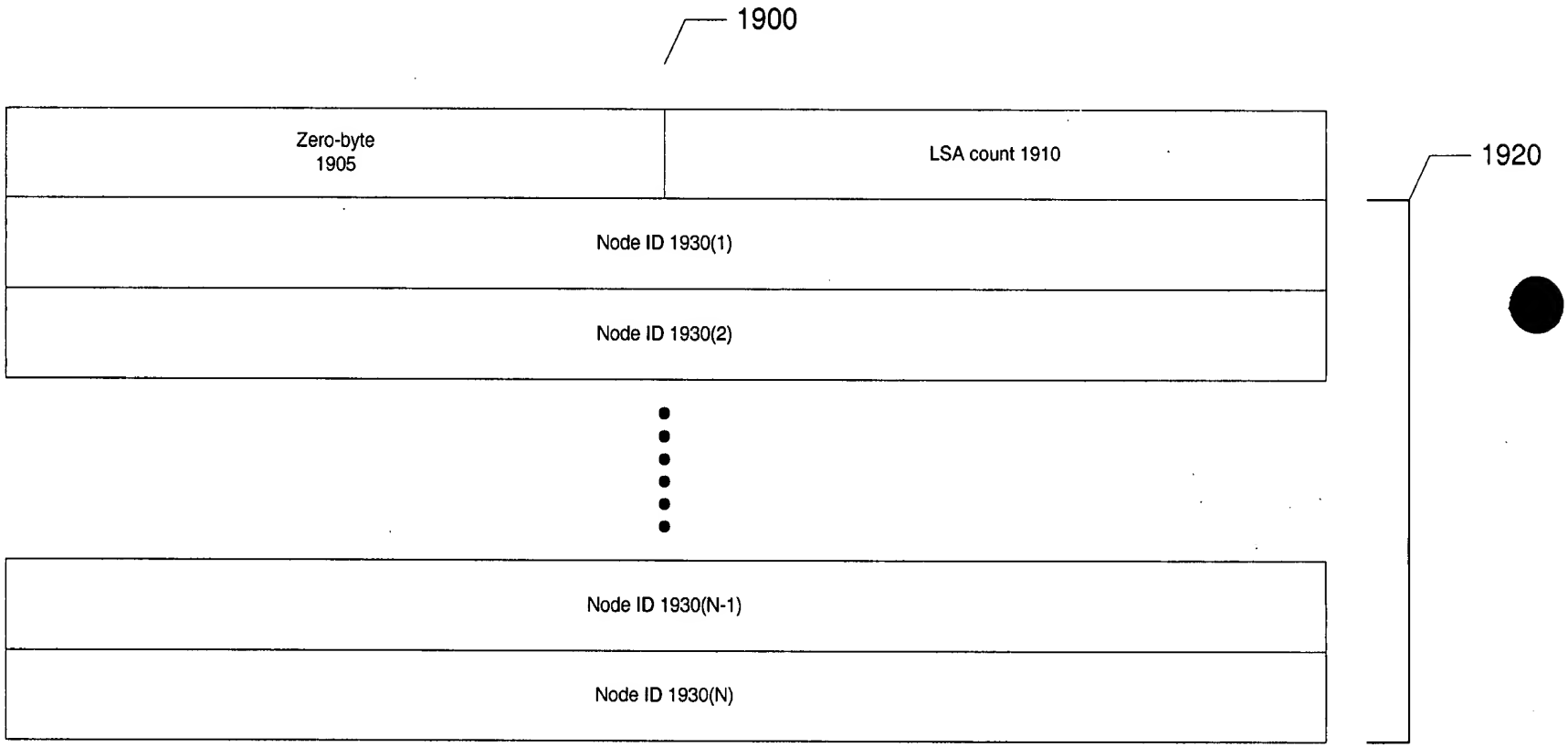
1900
Zero-byte
1905
LSA count 1910
1920
Node ID 1930(1)
Node ID 1930(2)
Node ID 1930(N-1)
Node ID 1930(N)

Fig. 19

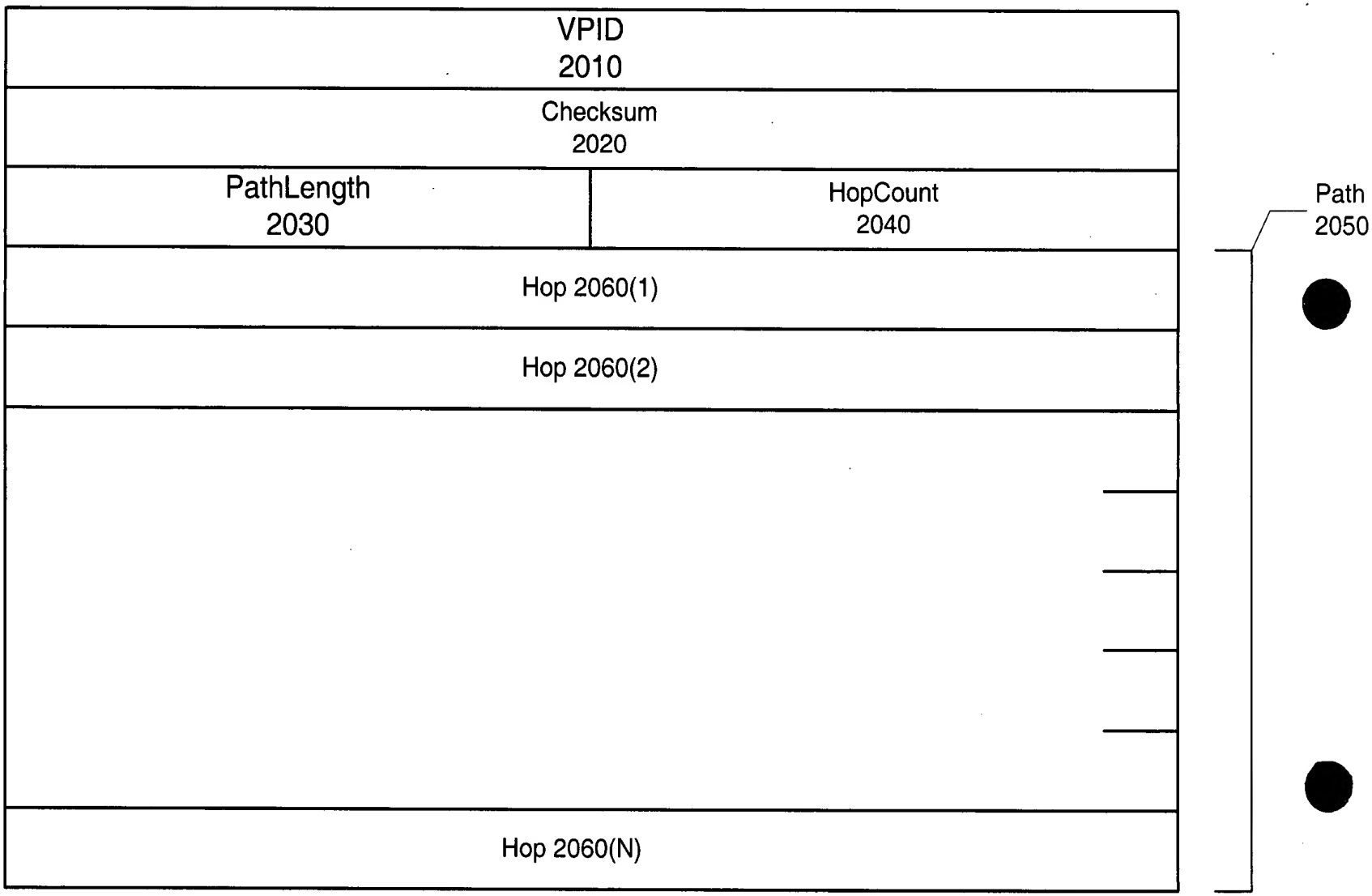
VPID
2010
Checksum
2020
PathLength
2030
HopCount
2040
Path
2050
Hop 2060(1)
Hop 2060(2)
Hop 2060(N)

Fig. 20

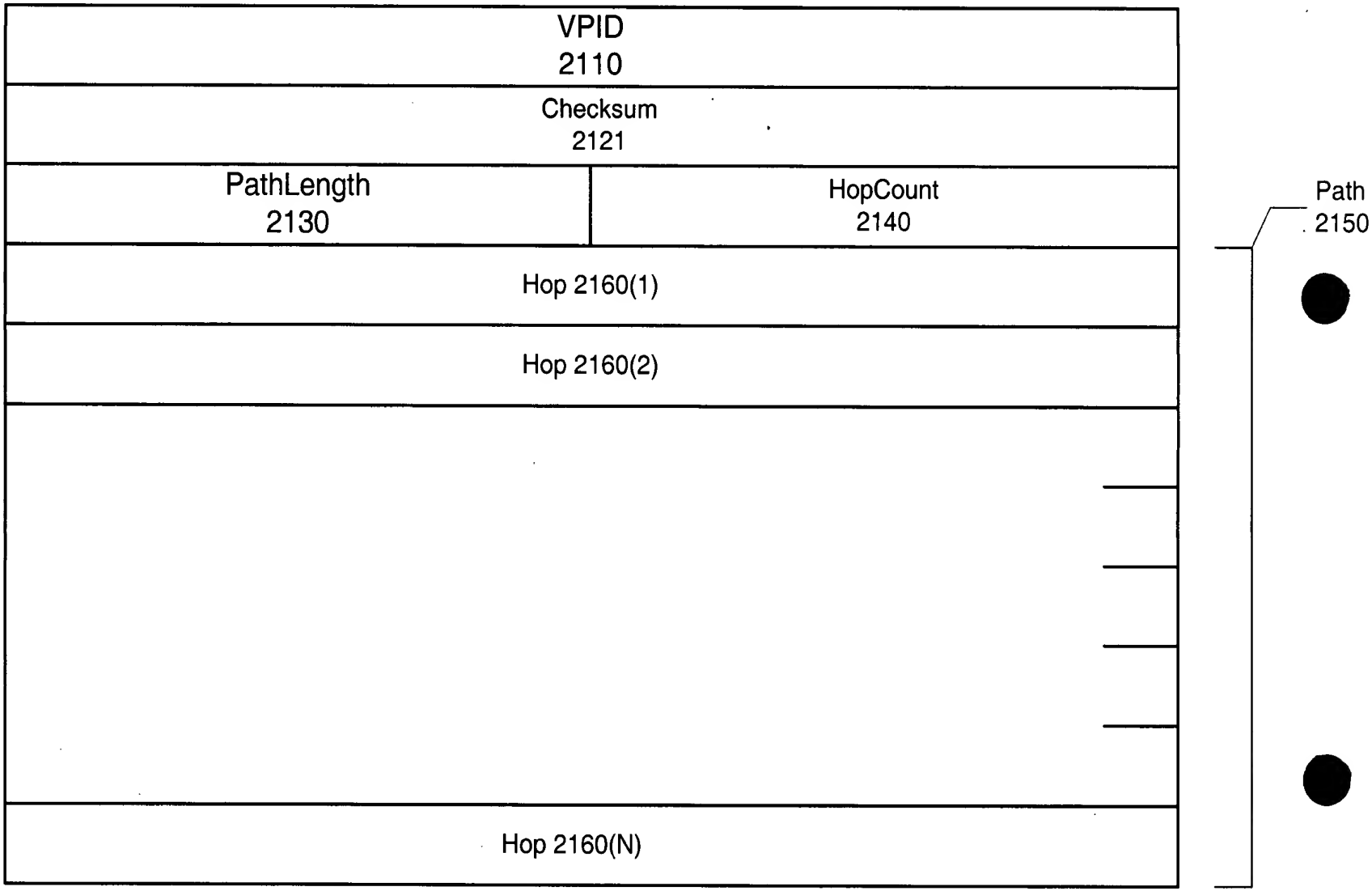
VPID
2110
Checksum
2121
PathLength
2130
HopCount
2140
Path
2150
Hop 2160(1)
Hop 2160(2)
Hop 2160(N)

Fig. 21

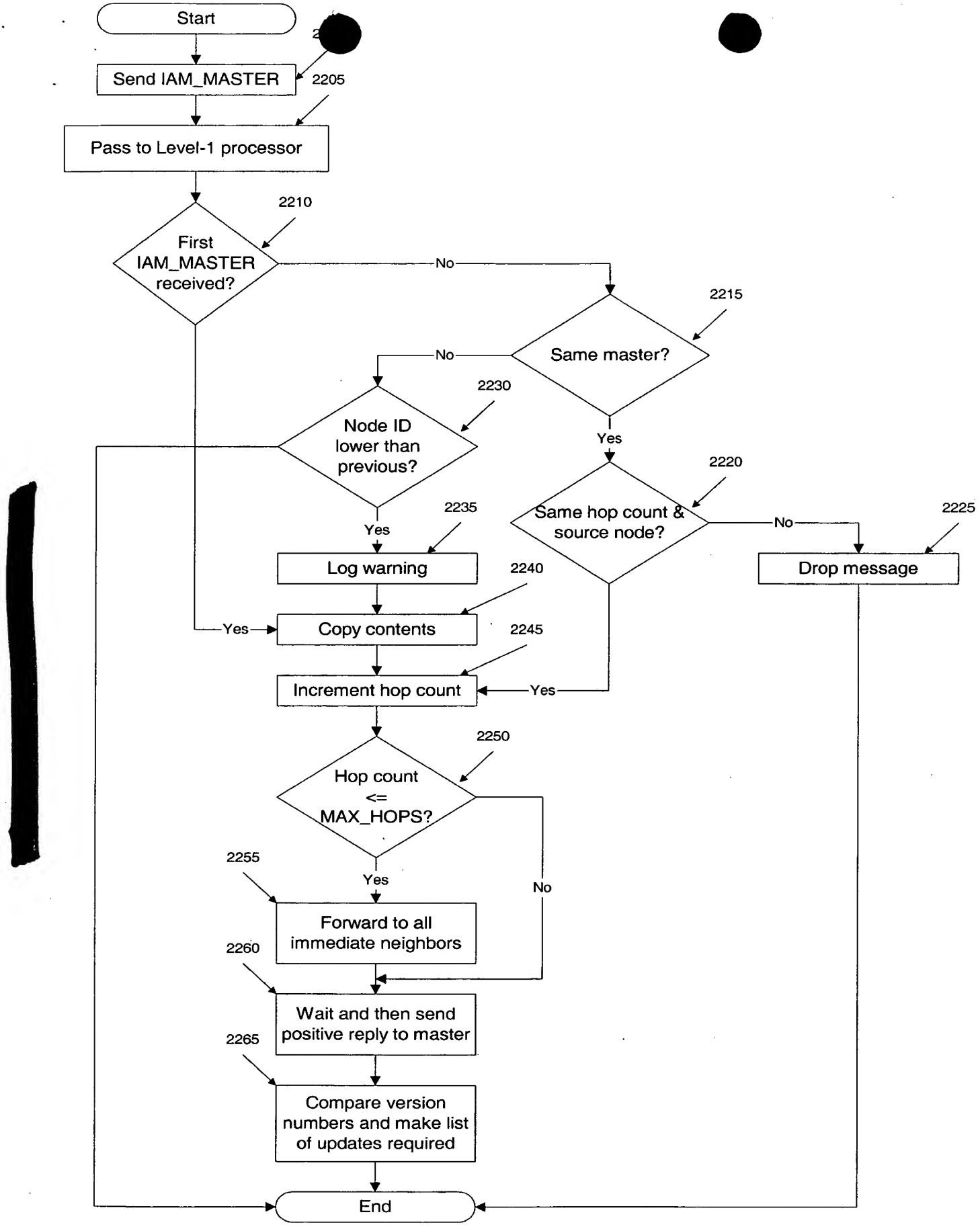
Start
Send IAM_MASTER
2205
Pass to Level-1 processor
2210
First IAM_MASTER received?
No
2215
Same master?
No
2230
Node ID lower than previous?
Yes
2220
Same hop count & source node?
No
2225
Drop message
Yes
2235
Log warning
2240
Copy contents
Yes
2245
Increment hop count
Yes
2250
Hop count <= MAX_HOPS?
Yes
No
2255
Forward to all immediate neighbors
2260
Wait and then send positive reply to master
2265
Compare version numbers and make list of updates required
End

Fig. 22

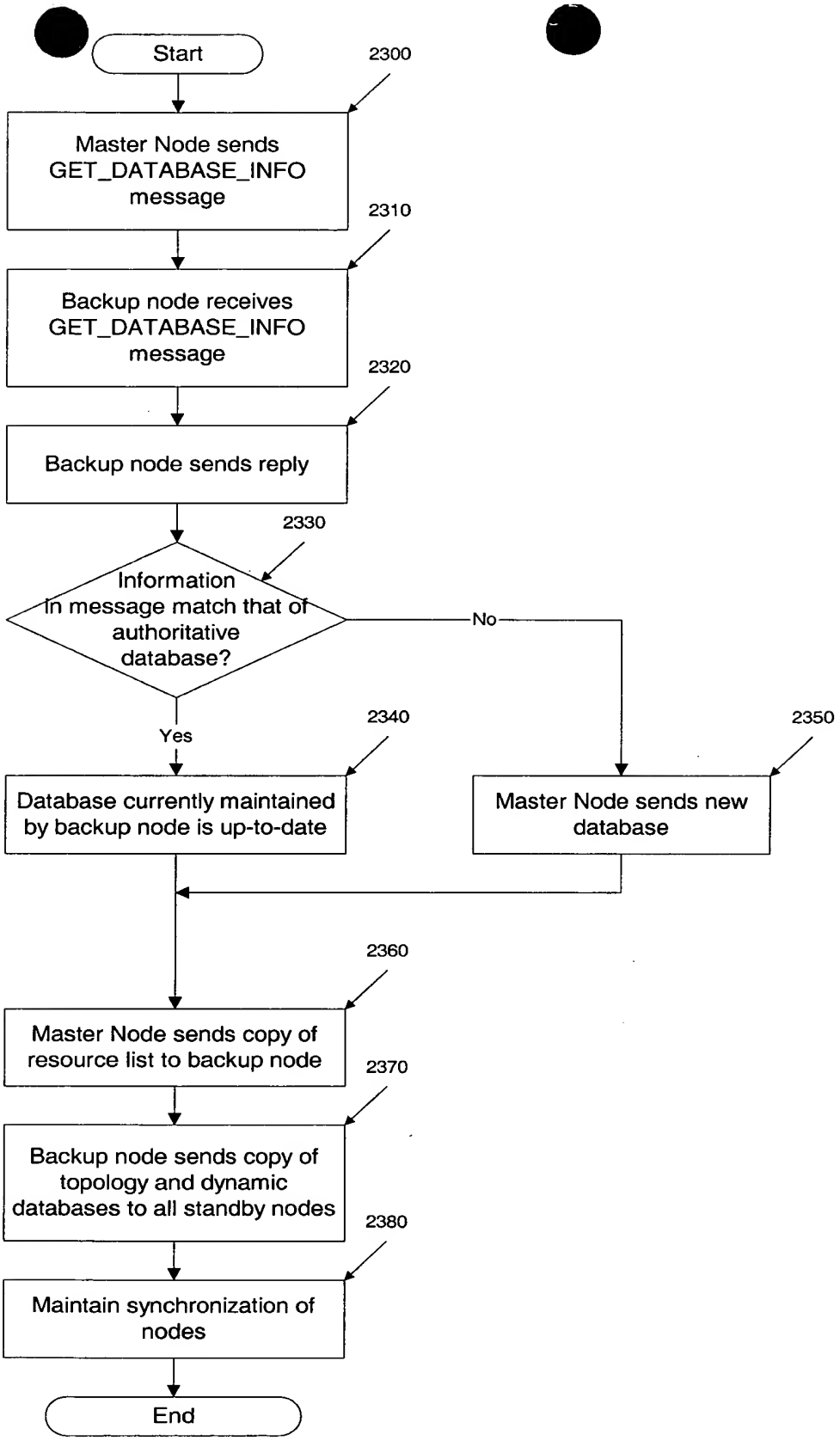
Start
2300
Master Node sends GET_DATABASE_INFO message
2310
Backup node receives GET_DATABASE_INFO message
2320
Backup node sends reply
2330
Information in message match that of authoritative database?
No
Yes
2340
Database currently maintained by backup node is up-to-date
2350
Master Node sends new database
2360
Master Node sends copy of resource list to backup node
2370
Backup node sends copy of topology and dynamic databases to all standby nodes
2380
Maintain synchronization of nodes
End

Fig. 23

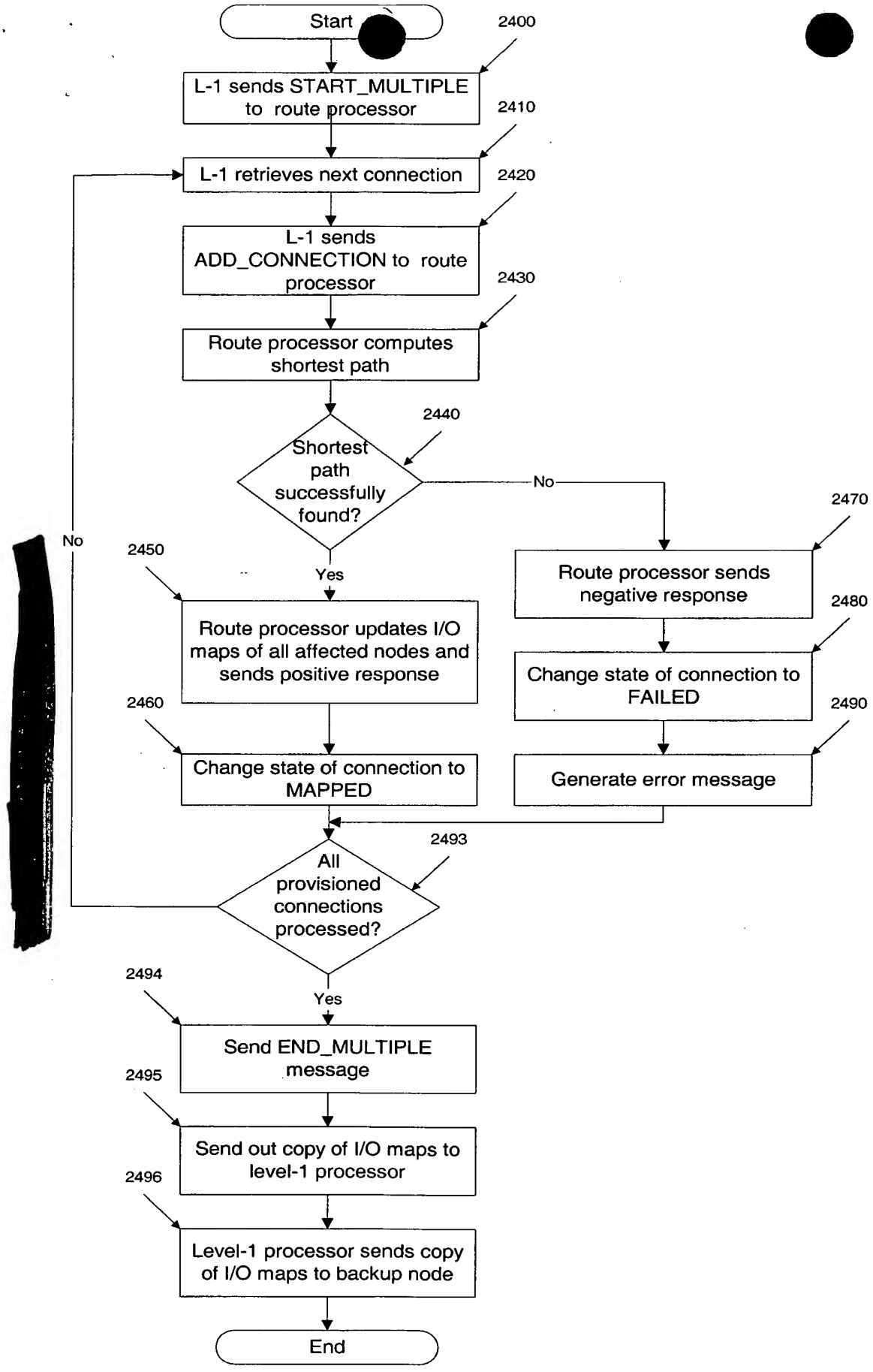
Start
2400
L-1 sends START_MULTIPLE to route processor
2410
L-1 retrieves next connection
2420
L-1 sends ADD_CONNECTION to route processor
2430
Route processor computes shortest path
2440
Shortest path successfully found?
No
Yes
2470
Route processor sends negative response
2480
Change state of connection to FAILED
2490
Generate error message
2450
Route processor updates I/O maps of all affected nodes and sends positive response
2460
Change state of connection to MAPPED
No
2493
All provisioned connections processed?
Yes
2494
Send END_MULTIPLE message
2495
Send out copy of I/O maps to level-1 processor
2496
Level-1 processor sends copy of I/O maps to backup node
End

Fig. 24

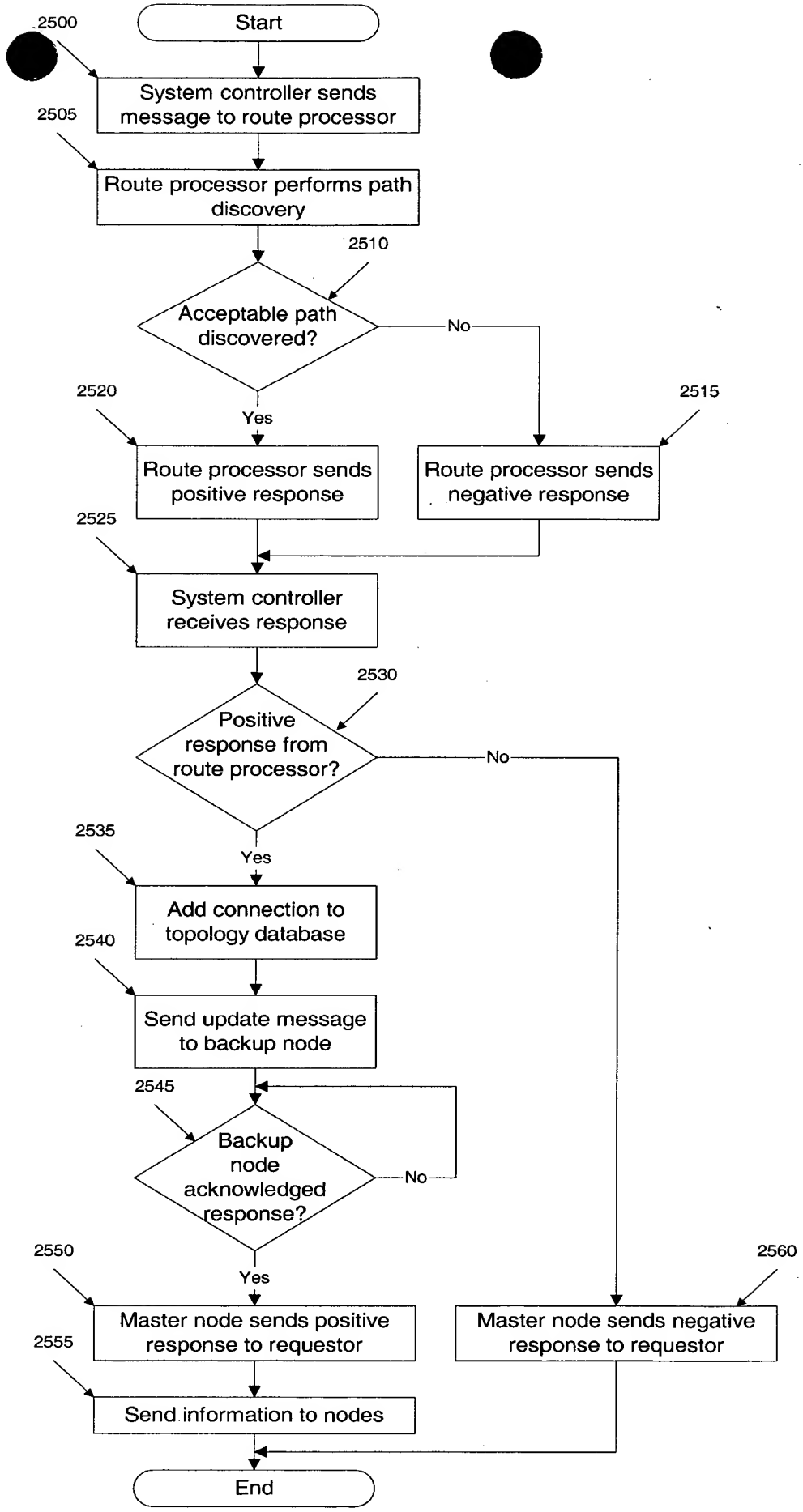
2500
Start
System controller sends message to route processor
2505
Route processor performs path discovery
2510
Acceptable path discovered?
No
Yes
2520
2515
Route processor sends positive response
Route processor sends negative response
2525
System controller receives response
2530
Positive response from route processor?
No
Yes
2535
Add connection to topology database
2540
Send update message to backup node
2545
Backup node acknowledged response?
No
Yes
2550
2560
Master node sends positive response to requestor
Master node sends negative response to requestor
2555
Send information to nodes
End

Fig. 25

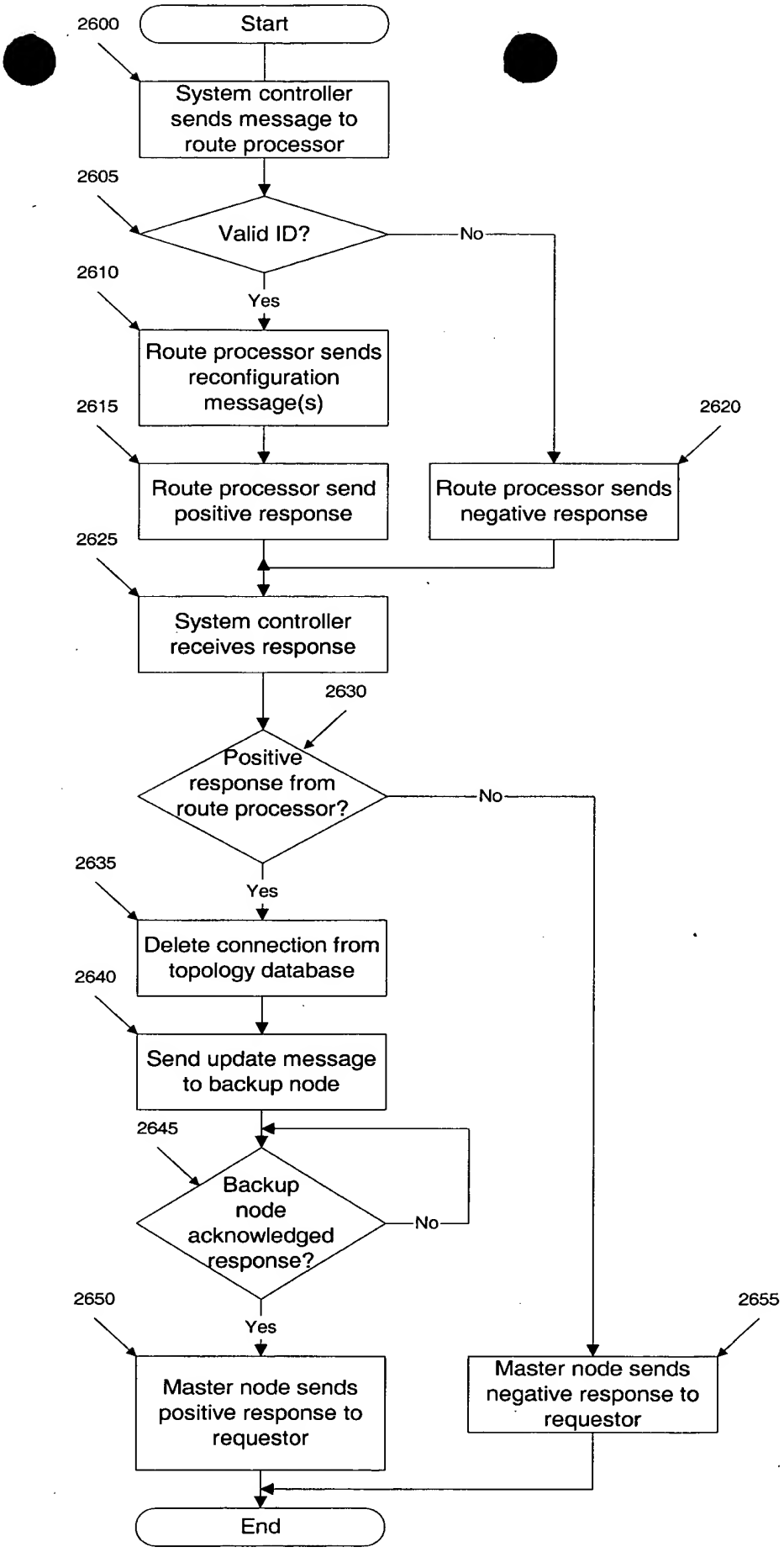
2600
Start
System controller sends message to route processor
2605
Valid ID?
No
Yes
2610
Route processor sends reconfiguration message(s)
2615
Route processor send positive response
2620
Route processor sends negative response
2625
System controller receives response
2630
Positive response from route processor?
No
Yes
2635
Delete connection from topology database
2640
Send update message to backup node
2645
Backup node acknowledged response?
No
Yes
2650
Master node sends positive response to requestor
2655
Master node sends negative response to requestor
End

Fig. 26

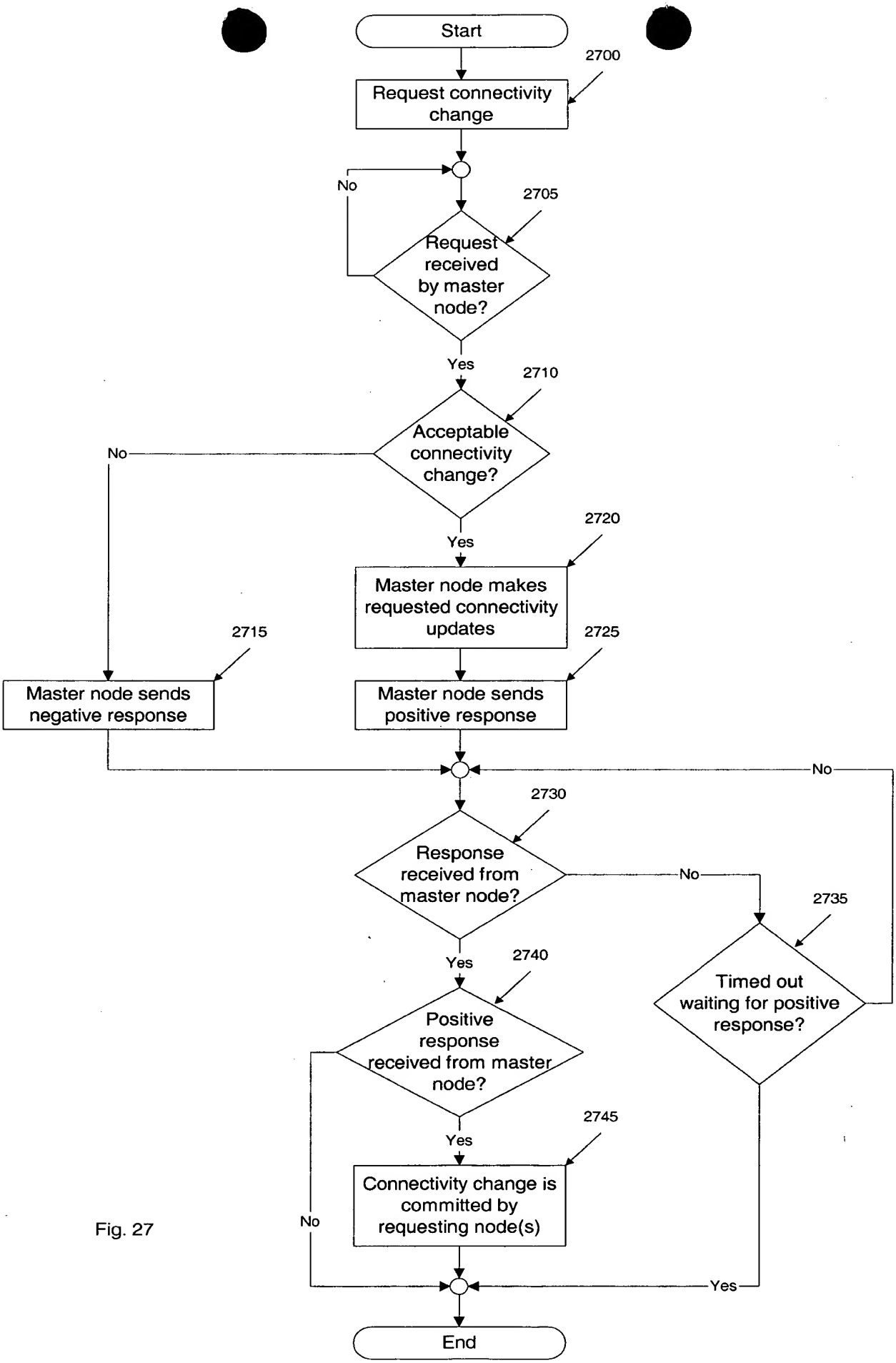
Start
2700
Request connectivity change
No
2705
Request received by master node?
Yes
2710
Acceptable connectivity change?
No
Yes
2720
Master node makes requested connectivity updates
2715
2725
Master node sends negative response
Master node sends positive response
No
2730
Response received from master node?
No
2735
Timed out waiting for positive response?
Yes
2740
Positive response received from master node?
Yes
2745
Connectivity change is committed by requesting node(s)
No
Yes
End

Fig. 27